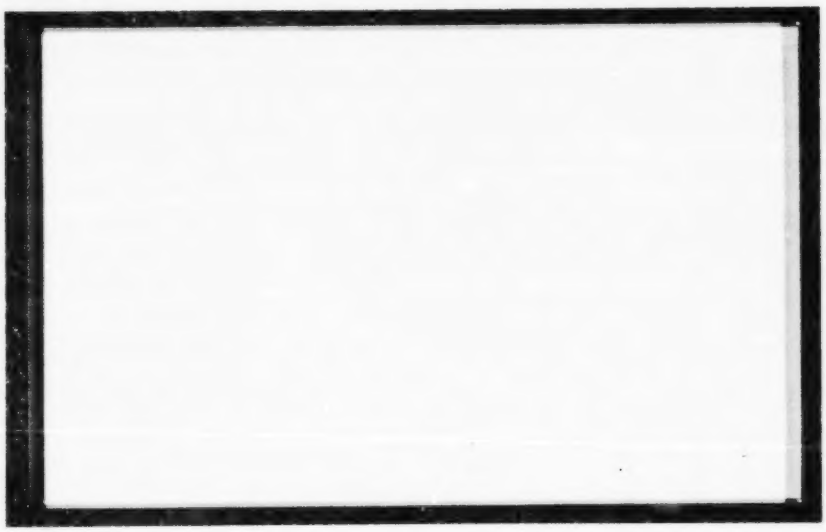
UNIVERSITÉ DE MONTRÉAL



CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

ROUTING AND SCHEDULING FOR THE TRANSPORTATION OF THE DISABLED PERSONS : THE ALGORITHM

by

Serge Roy, Jean-Marc Rousseau
Guy Lapalme and Jacques Ferland

Prepared for the Transportation Development Center,
Transport Canada - TP 5596 E - August 1984

Centre de recherche sur les transports - Publication #412A
Université de Montréal
June 1985

Dépôt légal - 3e trimestre 1985

The contents of this report reflect the opinions of the authors only, and in no way represent the policies or official views of the Transportation Development Centre, Transport Canada.

Également disponible en français sous le titre : "Élaboration des routes et horaires de véhicules pour le transport adapté : L'algorithme", #412F.

RESEARCH TEAM AND STEERING COMMITTEE

The research team from the Centre de recherche sur les transports de l'Université de Montréal was composed of :

- Luc Chapleau, M.Sc.,
- Jacques Ferland, Ph.D.,
- Guy Lapalme, Ph.D.,
- Claude Mallette, B.Sc.
- Jean-Marc Rousseau, Ph.D.,
- Serge Roy, M.Sc.

A steering committee was formed to evaluate the progress of the work and the final report. The members were :

Mr. Frank H. Ahlin
Senior Analyst Operations Research
Toronto Transit Commission

M. François Beaulieu
Directeur adjoint Soutien technique et administratif
Service du Transport
Commission de transport de la Communauté urbaine de Montréal

Mr. B.E. Hicks
Senior Program Manager
Road and Transportation Association of Canada

Mr. John Ryell
Transit Office
Ontario Ministry of Transportation and Communications

M. Michel Trudel
Agent de recherche, Service des études
Ministère des transports du Québec

Ms. Ling Suen
Senior Special Advisor
Transportation Development Center

ACKNOWLEDGEMENTS

The authors would like to thank the members of the steering committee, and the transportation authorities of Montréal, Sherbrooke and Toronto, for their collaboration in this project.

FORMULE DE DOCUMENTATION POUR RAPPORT

1. N° du rapport Transports Canada	2. N° de l'étude — CDT	3. N° de catalogue du destinataire
TP5596F	4712	

4. Titre et sous-titre	5. Date du rapport
Elaboration des routes et horaires de véhicules pour le transport adapté : l'algorithme	Août 1984
	6. N° du rapport de l'organisme exécutant

7. Auteur(s)	8. N° de dossier Transports Canada
Serge Roy, Jean-Marc Rousseau, Guy Lapalme, et Jacques Ferland	D1450-103-51

9. Nom et adresse de l'organisme exécutant	10. N° de dossier — ASC
Centre de recherche sur les transports Université de Montréal C.P. 6128 - Succursale "A" Montréal, Québec H3C 3J7	6 AG.T82-2-2511
	11. N° de contrat — ASC ou Transports Canada OAG82-00044

12. Nom et adresse de l'organisme parrain	13. Genre de rapport et période visée
Centre de développement des transports Transports Canada Bureau 2421, 1000 rue Sherbrooke Ouest Montréal, Québec H3A 2R3	Final
	14. Code de l'organisme parrain CDT 4712

15. Remarques additionnelles	16. Agent de projet — CDT
	S. Corbeil, L. Parent, T. Smith, C. Hirou

17. Résumé

Les principaux objectifs du projet étaient de développer un algorithme et des programmes informatiques pour confectionner des routes et des horaires de véhicules. Un objectif secondaire était la codification de problèmes tests pouvant servir à valider le logiciel développé et à d'autres études touchant le domaine du transport adapté. L'algorithme et les programmes informatiques furent améliorés successivement, à l'aide des tests effectués avec les données codifiées. Dans le système développé, un soin particulier est apporté à l'évaluation des distances entre les arrêts à visiter. Le dernier point nous a amenés à utiliser des AMF (Area Master Files), un ensemble de fichiers contenant diverses données géographiques. Dans l'algorithme de confection des routes et d'horaires, la qualité du service et la productivité des véhicules sont les critères considérés. Les résultats obtenus avec les tests effectués démontrent les avantages du système informatique comparativement à la méthode manuelle.

Le présent rapport décrit l'aspect technique d'un algorithme de confection de routes et d'horaires pour un système de transport semi-collectif pour personnes handicapées. L'objectif est de confectionner des routes et horaires qui maximisent le taux de productivité des véhicules en considérant une qualité de service fixée à chacune des demandes de transport. L'algorithme est en mesure de traiter des problèmes de grande taille.

18. Mots-clés	19. Diffusion
horaire, route, transport adapté	Nombre limité d'exemplaires disponibles au C.D.T.

20. Classification de sécurité (de ce rapport)	21. Classification de sécurité (de cette page)	22. Nombre de pages	23. Prix
Non classifié	Non classifiée	21	nil

REPORT DOCUMENTATION FORM

1. Transport Canada Report No.	2. TDC Project No.	3. Recipient's Catalogue No.
TP5596E	4712	

4. Title and Subtitle	5. Report Date
Routing and Scheduling for the Transportation of the Disables Persons : The Algorithm	August 1984
	6. Performing Organization Report No.

7. Author(s)	8. Transport Canada File No.
Serge Roy, Jean-Marc Rousseau, Guy Laplame, and Jacques Ferland	D1450-103-51

9. Performing Organization Name and Address	10. DSS File No.
Centre de recherche sur les transports Université de Montréal C.P. 6128 - Succursale "A" Montréal, Québec H3C 3J7	6 AG.T82-2-2511
	11. DSS or Transport Canada Contract No.
	OAG82-00044

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
Transportation Development Centre Transport Canada Suite 2421, 1000 Sherbrooke Street West Montreal, Quebec H3A 2R3	Final
	14. Sponsoring Agency Code
	TDC 4712

15. Supplementary Notes	16. TDC Project Officer
	S. Corbeil, L. Parent, T. Smith, C. Hirou

17. Abstract

The main objective of the project were to develop an algorithm and computer programs for the construction of vehicle routes and schedules. A secondary objective was the coding of test problems which could be used to validate the software developed or which could be used in other studies concerning the transportation of the handicapped. The algorithm and computer programs were successively improved with the help of tests carried out on the coded data. In the system developed, special attention was devote to the evaluation of the distance between stops to be visited. This led us to the use of the Area Master Files, a set if files containing various items of geographical data. In the route and schedule construction algorithm, quality of service and vehicle productivity are the two criteria considered. Test results demonstrate the advantages of the computer system over manual methods.

This report describes the technical aspects of the route and schedule construction algorithm for a semi-collective transportation system for the handicapped. The objective is to maximize vehicle productivity for a fixed level of service for each request.

18. Key Words	19. Distribution Statement
routing, scheduling, transportation of the disabled	A limited number of copies are available from the sponsoring agency

20. Security Classification (of this report)	21. Security Classification (of this page)	22. No. of Pages	23. Price
Unclassified	Unclassified	21	Nil

SUMMARY

This document describes the technical aspects of the route and schedule preparation algorithm for a semi-collective transportation system for the handicapped. This software was developed and tested by the Centre de recherche sur les transports of the Université de Montréal under a grant from the Road and Transportation Association of Canada (R.T.A.C.) And the Transportation Development Centre (TDC) of Transport Canada. This software may also serve as a tool to evaluate operating strategies.

Two types of problems related to this kind of transportation have been studied in the literature. The first problem involves the construction of routes to respond to a set of requests known in advance [2,5]. The second involves the integration of a new request into an already existing route [4]. In this paper, we propose a heuristic which can handle both problems in which the second problem merely becomes a special case of the first. This heuristic can solve large scale problems. In the first step, the objective is to maximize vehicle productivity with the quality of service fixed for each request. In the next step, the quality of service is maximized while maintaining vehicle productivity at the value obtained in the previous step.

TABLE OF CONTENTS

1. INTRODUCTION 1

2. THE DATA 2
   2.1 The requests. 2
   2.2 Driver duty 2
   2.3 The distance table. 3
      2.3.1 Location methods 3
      2.3.2 Estimation methods 3
      2.3.3 Tables types 4
   2.4 The speed table 4
   2.5 Dwell times 5
   2.6 The quality of service. 6

3. DEFINITION. 8
   3.1 The time window concept 8
   3.2 The "neighbour" concept 10
   3.3 A measure of affinity 11
   3.4 The horizon concept 15

4. THE INSERTION HEURISTIC. 17
   4.1 Route initialization. 17
   4.2 Determination of the next request to be inserted. 17
   4.3 Selection of the best route for a given request 18
   4.4 Insertion of a request into a route 18
   4.5 Phases of insertion 18
   4.6 The heuristic 19

5. POST-OPTIMIZATION. 20

6. REFERENCES 21

LIST OF FIGURES

Figure 1 : Speed function 5
Figure 2 : Time window, departure desired 8
Figure 3 : Time window, arrival desired 9
Figure 4 : Neighbouring request, case 1 10
Figure 5 : Neighbouring request, case 2 11
Figure 6 : The sequencing of two stops. 12
Figure 7 : Affinity request 12
Figure 8 : Sequencing request 13

1. INTRODUCTION

The number of handicapped persons using dedicated urban public transportation service is constantly increasing. This places pressure on the service to improve both its efficiency and accessibility. While there exist management information systems to help any administration in handling the complex task of reservation, scheduling and dispatching there is at present no adequate algorithm available to generate a good schedule from a large group of travel requests.

This document describes the technical aspects of the route and schedule preparation algorithm for a semi-collective transportation system for the handicapped. This software was developed and tested by the Centre de recherche sur les transports of the Université de Montréal under a grant from the Road and Transportation Association of Canada (R.T.A.C.) And the Transportation Development Centre (TDC) of Transport Canada. This software may also serve as a tool to evaluate operating strategies.

This type of transportation is expensive in terms of human and financial resources. The manual construction of routes and schedules for dozens of vehicles transporting hundreds of users is an extremely complex task. For example, The Montréal Urban Community Transit Commission (M.U.C.T.C.) has to construct routes for approximately fifty vehicles to serve an area of about 300 square kilometres.

Two types of problems related to this kind of transportation have been studied in the literature. The first problem involves the construction of routes to respond to a set of requests known in advance [2,5]. The second involves the integration of a new request into an already existing route [4]. In this paper, we propose a heuristic which can handle both problems in which the second problem merely becomes a special case of the first. This heuristic can solve large scale problems. In the first step, the objective is to maximize vehicle productivity with the quality of service fixed for each request. In the next step, the quality of service is maximized while maintaining vehicle productivity at the value obtained in the previous step.

There are two distinct techniques for solving route preparation problems for several vehicles; insertion heuristics and clustering methods. The originality of our heuristic lies in the use of both of these approaches. For a given level of service, we specify a time interval during with each stop must be made. A measure of affinity between a request and a route is defined as a function of the time windows and the distance between the stops involved. This measure is used to guide the parallel insertion of groupes of request. Two requests are placed in the same group if they have a common stop in terms of both time and space.

2. THE DATA

2.1 Requests

Each request involves the movement of an individual or group from a place of origin to a destination. A time for the service is specified corresponding to a desired departure or arrival time, depending on the type of request. The group to be transported may include handicapped persons who are ambulatory or in a wheelchair, and accompanying persons.

For each request (i) the following data is available :

o(i) : Place of origin indicator for request (i).
d(i) : Destination indicator for request (i).
NBAM(i) : Number of ambulatory handicapped persons.
NBCH(i) : Number of handicapped persons in wheelchairs.
NBCO(i) : Number of accompanying persons.
PRIORITE(i) : Quality of service to be provided.
TYPE(i) : Type of request (departure time required or arrival time required).[1]
HS(i) : Arrival or departure time required depending on the type of request (TYPE (i)).

2.2 Driver duty

A set of driver duties (or workday) must be input into the heuristic. These duties can be supplied by the user, or produced by HASTUS-macro, an algorithm for driver duty preparation (see Blais and Rousseau [1]). The use of this algorithm requires an estimate of the number of vehicles required for each period of the day, which can be obtained by examining the average demand profile. It is important to note that the heuristic may only use a subset of the duties proposed. Each route produced by the heuristic corresponds to either a duty or a task.[2]

For each driver duty (j) we have the following information :

NBP (j) : The number of tasks in workday j.

For each piece of work k in the workday j :

o(k) : Indicator of location where piece of work k begins.
d(k) : Indicator of location where piece of work k finishes.
HD(k) : Starting time for piece of work k.
HF(k) : Finishing time for piece of work k.

We also have the vehicle capacity assigned to this driver for the piece of work.

---

[1] The user can either require a specific departure time or arrival time, but not both.

[2] Duties for drivers in transit are in general made up of one or two tasks.

2.3 The distance table

Two stops correspond to each request : the origin and destination. For n requests, there are 2n stops to which correspond m different locations or points ($m \leq 2n$). The set of stops mapped onto a point depends on the location method.

2.3.1 Location methods

i. All stops having the same address are located at the same point.

ii. Stops are grouped together according to the distance between them. Each group corresponds to a point.

iii. The territory is divided into sectors, which can be determined in three different ways :
   - using postal codes,
   - using a grid,
   - manually.

   A point corresponds to each sector. This is the centroid of the sector.

2.3.2 Estimation methods

We define d(i,j), as the distance between point (i) and point (j). This distance may be calculated in various ways :

1) Let (x_i, y_i) and (x_j, y_j) be the coordinates of points i and j. The distance between i and j could be evaluated as a function of these coordinated. The simplest function (Manhattan distance) would be :

$$d_{ij} = |x_i - x_j| + |y_i - y_j|$$

The Euclidean distance (crow fly distance) would be :

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

There are several variants of these functions. These methods require that the points be geocoded.

2) The road network is represented by a graph where an arc represents a street segment and a node represents an intersection. Each point is associated with a node of the graph i.e. the points are located at intersections of the road network. The distance between the nodes is calculated using a shorthest path algorithm.

We have tested these methods. The latter one was adopted because if produces more accurate estimates using less computer time. In addition, the use of a network allows stops to be located automatically.

This comparison of the two methods also supplied new data on a much debated issue.

2.3.3 Distance tables

Because the estimation of the distance between two points is needed frequently in the algorithm and because this is a time consuming task, we chose to work with distance tables. We consider two types of tables :

A complete table : The distance between each pair points is known.
A partial table : For a given point, the distance to neighboring points only is known.

The type of table constructed, and the location estimation methods used are chosen according to the number of points, and the surface area of the territory to be covered. For m points, the dimension of the complete table is m(m+1)/2.

d(i,j) : The distance between point (i) and point (j), j i
d(i,i) : The error due to the method of location; the average distance between two stops assigned to the same point.

2.4 The speed table

The system allows the user to define speed as a function of distance. This approach is more realistic : for short distances, there is always a fixed time required for acceleration and deceleration, and for long distances, higher speeds are possible because of the ability to use fast lanes.

The function is specified using six parameters :

VITMIN : Minimum speed. (metres/minute)
VITMOY : Average speed. (metres/minute)
VITMAX : Maximum speed. (metres/minutes)
d1 : The distance at which the average speed is applied (metres).
d2 : The distance at which the speed becomes higher than the average speed. (metres)
d3 : The distance above which the maximum speed is applied. (metres)

Using these parameters, the following function is applied :

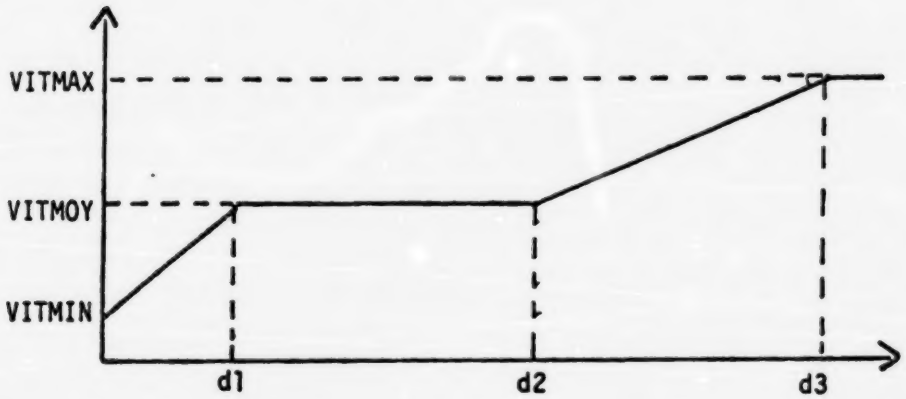


Figure 1 Speed function

For a distance d, our function gives a speed v(d).This allows us to calculate the travel time t=d*v(d). It is important that a given travel time corresponds to only one distance. To ensure that the distance can be found from the time, the speed function must satisfy the following condition.

Any straight line passing through the origin must intersect the speed function at only one point. This constraint ensures that a given time corresponds to a single distance, enabling this distance to be found.

2.5 Dwell times

For each stop, a handling time is calculated. This time is a function of the number and type of handicapped included in the request corresponding to this stop. The user must apply the following values :

CHAISEMAN : Time required for handling a wheelchair.
AMBUMAN : Time required for handling an ambulatory handicapped person.

If the request involves several people, then the complete handling time for the first individual is included plus half the basic handling time for each additional person. This method of calculation takes into account the fact that the driver has alighted and the doors of the vehicle are open.

If (i) is a stop for request "d", the handling time at stop (i) is thus given by :

$$TM(i) = \begin{cases} (NBCH(d)+1)/2.0 * CHAISEMAN + NBAM(d)/2*AMBUMAN & \text{if } NBCH(d) > 0 \\ (NBAM(d)+1)/2.0 * AMBUMAN & \text{if } NBCH(d) = 0. \end{cases}$$

The user may specify the handling time at the origin or destination of a request to take into account special cases. If no time is specified, the function given above is used.

The user may also input times allowed to the driver at the beginning and end of the task

DEPGAR : Time allowed to the driver for departure from the garage.
ARRGAR : Time allowed to the driver for arrival at the garage.

2.6 The quality of service

In selecting the routes, the heuristic fixes for each request (i):

HD(i) : The time at which the vehicle leaves the origin of request (i).
HA(i) : The time at which the vehicle arrives at the destination of request (i).

Between the departure and arrival of passengers for a given request, the vehicle may visit other stops to serve other requests, thus generating excess travel time for the passengers of the original request.

ERT(i) : Excess ride time for request (i)
ERT(i) = HA(i) - HD(i) - t(o(i),d(i)).

For each request (i) we added the request type (departure or arrival time desired) as well as the precise time HS(i) that the service is required. We define :

DTD(i) : Deviation from desired service time.

$$DTD(i) = \begin{cases} HD(i)-HS(i) & \text{if type (i) = Departured required} \\ HS(i)-HA(i) & \text{if type (i) = Arrival required.} \end{cases}$$

The transport company specifies the constraints on the quality of service provided to users. This quality measure will depend on the priority of the request. The quality of service provided to each priority is specified by the following parameters : the maximum deviation from the desired time allowed and the maximum excess travel time permitted.* In addition, the maximum time a passenger may spend in the vehicle can also be specified.

DM(j) : Maximum deviation from desired service time for PRIORITE(j).
1) DTD(i) < DM(PRIORITE(i))

EM(j)* : Maximum excess ride permitted for PRIORITE(j).
2) ERT(i) < EM(PRIORITE(i))

VOYAGEMAX : Maximum travel time.
3) HD(i)-HA(i) < VOYAGEMAX.

* The excess ride time could be formulated in a more complex way as a percentage of the duration of a direct trip. A minimum level of excess ride could also be defined below which excess travel is not penalized. For example, if excess ride time is set at 100% of travel time, with a minimum of 10 minutes, then for a total travel time of less than 10 minutes, 10 minutes of excess ride time is allowed while for a travel time of 20 minutes, 20 minutes of excess ride time is allowed.

3. DEFINITION

3.1 The time window concept

Each request involves two stops i.e. an origin and a destination. For each of these stops, we define a time window. This is the interval of time within which the stop must be visited. These time windows enable the constraints on the deviation from the desired service time to be satisfied, and they help restrict excess travelling. However, they do not necessarily force the constraints on maximum excess travel time to be respected. When working with time windows, the excess travel constraint is relaxed. The time windows for a request are a function of the quality of service corresponding to the priority rating of the request.

For each stop (k) we define :

PLUSTOT(k) : The earliest time the vehicle may arrive at stop (k).

PLUSTARD(k) : The latest time the vehicle may arrive at stop (k).

We now show diagramatically how the time windows are determined for the stops included in a request depending on the type of request (i.e. departure or arrival desired).

Consider a request (i) with origin o(i), destination d(i) and type TYPE (i). If TYPE (i) = departure desired

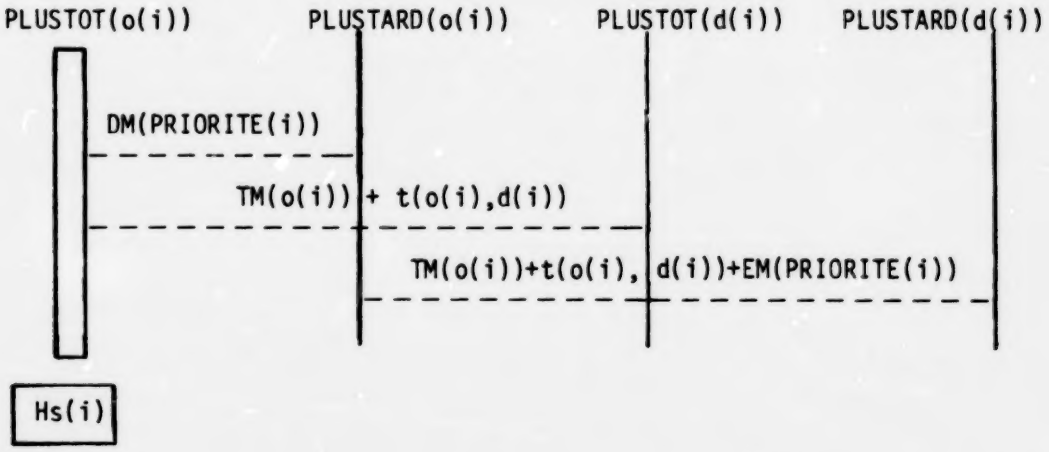


Figure 2 Time window, departure desired

If TYPE (i) = arrival desired

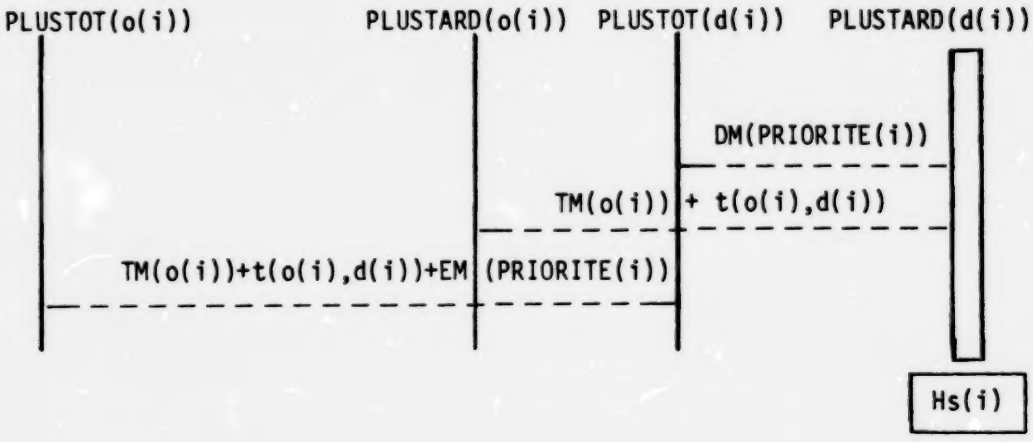


Figure 3 Time window, arrival desired

By defining the time windows in this way, we respect the constraint on the maximum deviation from the desired service time. The constraints on the maximum excess travel time and on the maximum total travel time are relaxed. These excess travel constraints become as follows :

$$4)\ ERT(i) < \begin{cases} EM(PRIORITE(i)) + DM(PRIORITE(i)) - (HD(i) - HS(i)) \\ \qquad \text{si TYPE (i) = departure required} \\ EM(PRIORITE(i)) + DM(PRIORITE(i)) - (HS(i) - HA(i)) \\ \qquad \text{si TYPE (i) = arrival required} \end{cases}$$

$$5)\ HD(i)-HA(i) < \begin{cases} VOYAGEMAX + DM(PRIORITE(i)) - (HD(i) - HS(i)) \\ \qquad \text{Si TYPE (i) = departure required} \\ VOYAGEMAX + DM(PRIORITE(i)) - (HS(i) - HA(i)) \\ \qquad \text{Si TYPE (i) = arrival required} \end{cases}$$

The values of EM(PRIORITE(i)) and/or DM(PRIORITE(i)) could be altered in order to respect the constraints specified by the transport company. But this approach would not allow a request to be satisfied at the limits on the deviation from the desired service time and at the limit of excess travel time. This would leave little latitute to the heuristic. It is better to initially relax the constraints on excess travel and on total travelling and then to come back to then during the post optimization phase, moving those requests which do not satisfy the constraints.

3.2 The "neighbour" concept

In most methods proposed in the literature [2, 5], the problem with several vehicles is tackled by applying a clustering technique. The multiple vehicle problem is reduced to several problems with one vehicle by determining the request to be handled by the same vehicle. The difficulty with this approach lies in the construction of clusters. Although it is not easy to identify all the requests to be handled by a single vehicle, it is however relatively easy to see that certain requests will be handled by the same vehicle. Requests with stops which are close to one another in both space and time are likely to use the same vehicle. This led us to define the concept of a mini cluster or group of neighbouring requests.

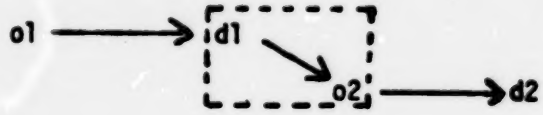
We define the constant :

VOISINMAX : The maximum distance allowed between two stops related to two different requests for the requests to be considered as "neighbours".

A necessary condition for two requests to be neighbours is that one involves a stop (ℓ) and the other a stop (k) such that

1) plustard(k)-plustot(ℓ) > t(ℓ,k) < VOISINMAX.

If stops (k) and (ℓ) are of different types, i.e. one is an origin and the other a destination, then the necessary condition is also sufficient. An example of a possible case is illustrated below.



d1 corresponds to stop (ℓ) and o2 corresponds to stop (k). If condition (1) is satisfied, the requests are considered to be neighbours. The position of the other two stops is noj relevant here.

Figure 4 Neighbouring request, case 1.

If stops (ℓ) and (k) are of the same type, (i.e. both are origins, or both are destinations), condition (1) is not sufficient and the position of the other two stops involved must be considered to ensure that the requests are going in the same direction. The additional condition required is illustrated by an example.

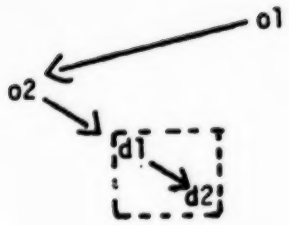


Figure 5 Neighbouring request, case 2

Suppose that d1 and d2 sastisfy condition (1). We require in addition that :

2) $t(o1, o2) + t(o2,d1) < 1.4 \cdot t(o1,d1)$.

The constant 1.4 is a parameter which approximately defines an ellipse around the stops. This value was determined from tests carried out on a school transportation problem.

3.3 A measure of affinity

We wish to define a measure of affinity between a request and a route to guide the parallel insertion algorithm. This measure must reflect the ease with which the request can be inserted into the route. The request has two stops : an origin and a destination. The precedence relationship between the origin and the destination must be respected. Each stop in the request and on the route has an associated time window. These time windows are adjusted to take into account the effect of other stops on the same route. Thus, it is easy to see that knowledge of the distances separating stops associated with a request from stops on a route is insufficient to determine this measure of affinity because of the need to consider "time" dimension associated with the time windows. Our measure must therefore integrate both "space" and "time" dimensions.

We first introduce a method of checking whether the sequencing of any stops is feasible. The same idea is used to check the feasibility of linking two requests. For a given request and route, if there exists a feasible link between the request and another request already inserted into the route, then our measure provides an indication of the detour caused by this link. Otherwise, (i.e. when there is no request already in the route which can be linked with the given request), then the measure takes a very large value indicating that the insertion of the request into the route is infeasible. Thus, the lower the value of the measure, the greater the affinity between the route and the request and the easier will insertion be.

Let (i) and (j) be two stops with time windows : [plustot(i), plustard(i)] and [plustot(j),plustard(j)] respectivety. Let t(i,j) be the time required to go from (i) to (j).

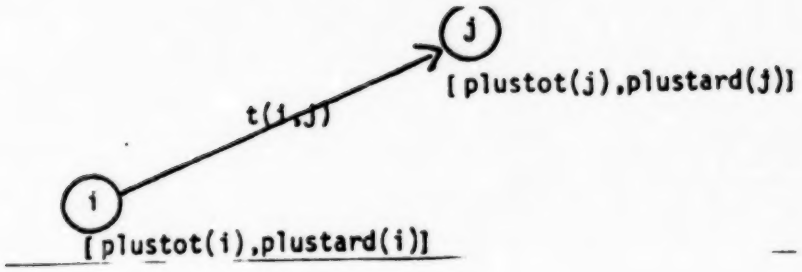


Figure 6 The sequencing of two stops

Considering these twop stops, the sequence (i) → (j) is feasible if we have

$$PLUSTARD(j)-PLUSTOT(i)-t(i,j) > 0$$

3.3.1 The sequencing of two requests

In order to visualize the problem, refer to the following diagram. The request shown by the dotted lines is our reference request, and this is considered in relation to four other requests with their time windows.

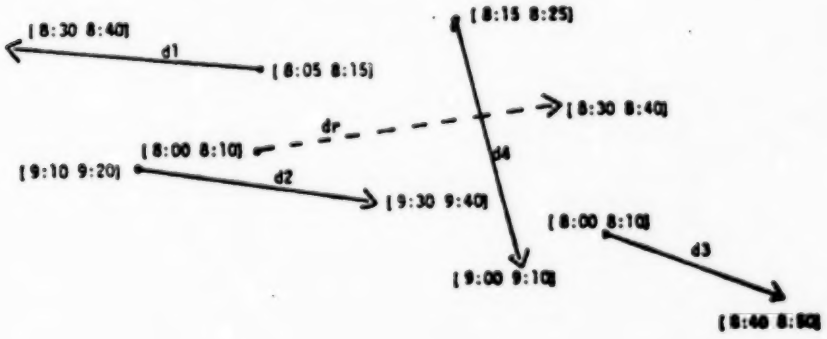


Figure 7 Affinity request

Which of the requests d1, d2, d3 and d4 has greatest affinity with the reference request (dr)? Request d1 is the opposite direction. Request d2 occurs too late and request d3 seems too far away. If the sequencing of dr and d4 given by o(dr) → o(d4) → d(dr) → d(d4) is examined, it can be seen that request d4 seems to have the greater affinity with dr.

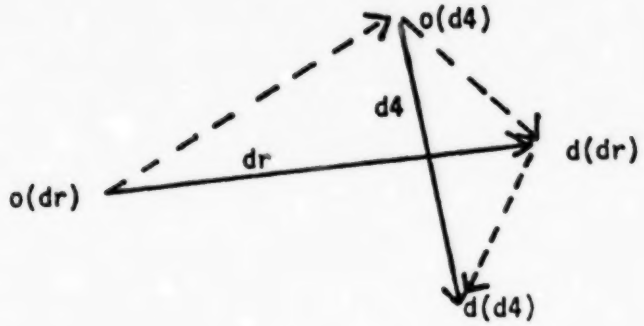


Figure 8 Sequencing request

Note that the affinity between two requests depends on the particular sequencing considered. Given the precedence relations, there are six ways of linking two requests. For example, for requests (k) and (ℓ) we may have

1) $o(k) \rightarrow d(k) \rightarrow o(\ell) \rightarrow d(\ell)$
2) $o(\ell) \rightarrow d(\ell) \rightarrow o(k) \rightarrow d(k)$
3) $o(k) \rightarrow o(\ell) \rightarrow d(k) \rightarrow d(\ell)$
4) $o(k) \rightarrow o(\ell) \rightarrow d(\ell) \rightarrow d(k)$
5) $o(\ell) \rightarrow o(k) \rightarrow d(\ell) \rightarrow d(k)$
6) $o(\ell) \rightarrow o(k) \rightarrow d(k) \rightarrow d(\ell)$

Based on the same principle used above to check on the feasibility of sequencing two stops, constraints can be defined to check on the feasibility of any given two requests.

In general, we have the sequence :

$$a \rightarrow b \rightarrow c \rightarrow d \ .$$

This involves a succession of three sequences of two stops. Each of these must be feasible for the total sequence to be valid. A sequencing of two requests is feasible if it satisfies the following three constraints :

1) The first constraint ensures the feasibility of the sequence :

$a \rightarrow b$

$$\text{Min}\{\text{PLUSTARD}(b), \text{Min}\{\text{PLUSTARD}(c), \text{PLUSTARD}(d) - t(c,d)\} - t(b,c)\}$$
$$- \text{PLUSTOT}(a) - t(a,b) > 0.$$

2) The second ensures the feasibility of the sequence :

$b \rightarrow c$

$$\text{Min}\{\text{PLUSTARD}(c), \text{PLUSTARD}(d) - t(c,d)\} - \text{Max}\{\text{PLUSTOT}(b), \text{PLUSTOT}(a) + t(a,b)\}$$
$$- t(b,c) > 0.$$

3) The third ensures the feasibility of the sequence

$$c \rightarrow d$$

$$PLUSTARD(d)-Max\{PLUSTOT(c),Max\{PLUSTOT(b),PLUSTOT(a)+t(a,b)\}+t(b,c)\}$$
$$-t(c,d) > 0.$$

For the 4th and 6th ways of linking the two requests, $b \rightarrow c$ is automatically feasible so constraint (2) does not have to be checked and constraints (1) and (3) can be simplified to give :

1) $Min\{PLUSTARD(b),PLUSTARD(d)-t(c,d)-t(b,c)\}-PLUSTOT(a)-t(a,b) > 0$

3) $PLUSTARD(d)-Max\{PLUSTOT(c),PLUSTOT(a)+t(a,b)+t(b,c)\}-t(c,d) > 0$

For the 1st and 2nd ways of linking the requests $a \rightarrow b$ and $c \rightarrow d$ are automatically feasible, so constraints (1) and (3) do not have to be checked and constraint (2) can be simplified to give :

2) $plustard(c)-plustot(b)-t(b,c) > 0.$

For the third and 5th ways of linking the requests, the constraints must be used in their most general form.

3.3.2 A measure of affinity between a request and a route

Consider a request (d) and a route (p). Using the constraints of section 3.3.1 we determine feasible links between the request (d) and the requests in the route (p). For each feasible link we estimate the detour for the route. This depends on the particular sequencing of stops considered.

For links of the 1st and 2nd type, which involve the movement t(b,c) of a possible empty vehicle we use as an estimate of the detour :

$$1.5 * t(b,c) + t(c,d).$$

This penalizes empty running.

For links of types 3, 4, 5 and 6, we use as an estimate :

$$t(a,b)+t(c,d).$$

For these sequences, t(a,b) corresponds to the time required to move from one origin to the other and t(c,d) is the time to go from one destination to the other.

We define a matrix with elements :

INSERT(d,p) : the measure of affinity between (d) and route (p).

INSERT(d,p) = { minimum detour over all feasible links between request (d) and the requests of route (p) - if there is at least one feasible link
MAXMESURE - if there is no feasible links

Thus, INSERT(d,p) indicates either that insertion of the request (d) into the route (p) is infeasible, or that the insertion is feasible and in this case, the function also provides an estimate of the detour involved in inserting request (d) into route (p). With this measure, it is now possible to estimate the cost of an insertion.

It is important to note that the value of INSERT(d,p) can change if a new request (j) is inserted into route (p). For, this insertion may modify the time windows of the stops on route (p). After each insertion of a request into a route, the time windows of the stops are adjusted. If for a stop (k) on a route we have a time window [PLUSTOT(k), PLUSTARD(k)], we obtain a new time window [PLUSTOT*(k),PLUSTARD*(k)] after inserting request (j) into the route. PLUSTOT*(k) will be the earliest time at which it is possible to arrive at stop (k) given the time windows on stops preceding (k). PLUSTARD*(k) is the latest time at which the vehicle may arrive at stop (k) to be able to respect the time window of the stops subsequent to stop (k). After insertion of a new request into the route we obtain :

$$PLUSTOT^*(k) \geq PLUSTOT(k)$$

$$PLUSTARD^*(k) \leq PLUSTARD(k).$$

3.4 The horizon concept

In the TAXI system ("A REALISTIC ADVANCE RESERVATION DIAL-A-RIDE ALGORITHM" [6], the requests which are known in advance and which are used to construct the routes, are classified according to the potential difficulty of inserting them into a route. They are then inserted one at a time in order of classification.

The order of insertion of requests is independent of the routes under construction. With our approach, we have a measure which reflects the difficulty of inserting a request into a route and which takes into account the location of the routes. We use this measure to determine the order of insertion of the requests. As an afternoon request does not influence the insertion of a morning request, it is useless to consider

all the requests for a given day at once. We therefore consider at the same time only requests having their desired service time within a given horizon. This horizon is determined by two values :

INFHORIZON : The starting time of the horizon corresponds to the earliest desired service time amongst all requests not yet inserted.

SUPHORIZON : The ending time of the horizon corresponds to the desired service time of the next request to enter into the horizon.

The management of the horizon is based on several values :

DEMHORIZON : The number of requests presently in the horizon.

HORIZONMIN : The minimum time interval allowed for the horizon.

MAXDEMHOR : The maximum number of requests allowed within the horizon.

The horizon moves forward in time by adding the next request as long as :

(SUPHORIZON-INFHORIZON) HORIZONMIN and DEMHORIZON $\leq$ MAXDEMHOR

In this way, the horizon advances gradually in time.

For all requests within the horizon we have measures of affinity with each request in the routes. These measures will determine the order of insertion of the requests. This continual adjustment of the horizon ensures that no requests are left behind without being inserted.

4. THE INSERTION HEURISTIC

The heuristic is based on a parallel insertion technique. All routes are constructed at the same time. After the initialization of a few routes, requests are inserted into them, or as needed new routes can be initialized.

We construct groups with the uninserted requests by using neighborhood links existing between the requests. These groups are used to initialize several routes at the beginning and to determine the next requests to be inserted.

Before presenting the heuristic, we describe in detail the following aspects of the algorithm :

4.1) How routes are initialized at the beginning.

4.2) How the next requests to be inserted are choses.

4.3) How the best route for a given request is selected.

4.4) Where the origin and destination of a request are inserted into a route.

4.1 Route initialization

For each group with a sufficiently large number of requests, a route is initialized and an attempt is made to insert all the requests of the group. The route is initialized by finding an available piece of work which can carry out the earliest request in the group. The piece of work chosen is that for which the least possible time is lost between the departure time of the piece of work and the time at which the vehicle must pick up the earliers request of the group under the following priority order :

1) Pieces of work for workdays already initiatized (days with a piece of work already initialized).

2) First pieces of work for non-initialized workdays.

3) Other pieces of work.

4.2 Determination of the next requests to be inserted

The request within the horizon which has the greatest number of uninserted neighbours is identified. If this request (centroid) has sufficient neighbours, it will be chosen along with its neighbours for an attempt at insertion into a route. If the group formed by these requests is small, the request within the horizon having the best affinity measure with the routes will be chosen.

4.3 Selection of the best route for a given request

The matrix INSERT(d,p) provides an indication of the detour caused by the insertion of a request d if this insertion is feasible. Thus, the best route for d is the route p* such that

$$INSERT(d,p^*) = \min_{p} \{INSERT(d,p)\}$$

4.4 Insertion of a request into a route

A request is to be inserted into a route. An insertion is defined by a pair of stops (GORI,GDES).

GORI : The stop in the route preceding the place where the origin of the request is to be inserted.

GDES : The stop in the route preceding the place where the destination of the request is to be inserted. (If GDES=GORI the destination of the request will be inserted immediately after the origin.)

A pair (GORI,GDES), with GORI < GDES, forms a feasible insertion if, when the time windows of the stops in the route after the insertion are recalculated, none of them closes up completely, i.e. we still have:

PLUSTOT*(k) < PLUSTARD*(k) for all stops k in the route.

Each feasible insertion involves a corresponding detour of the route and some inconvenience to clients. A feasible insertion may be chosen according to different criteria :

1) That which involves the minimun detours;
2) That which causes the least inconvenience.

4.5 Phases of insertion

Phase 1

The request "d" is to be inserted. Phase 1 of insertion involves finding the best route to receive request "d". Requests are evaluated in increasing order of the measure INSERT(d,p). The best route for "d" is p* such that

$$INSERT(d,p^*) = \min \{INSERT(d,p\} \text{ for all } p.$$

Phase 2

Requests are to be inserted into route (p). Phase 2 of insertion involves finding the best requests to be inserted into route (p). The best requests for this route (p) are uninserted requests which are neighbour of a request in route (p) or requests with a good measure of affinity with route (p), i.e. with a small value of INSERT(d,p).

4.6 The heuristic

1) Sort the requests into desired service time order.

2) Read the requests to initialize the horizon and to construct groups of neighbouring requests.

3) Initialize one or several routes with groups of neighbouring requests.

4) WHILE (all requests not inserted).

BEGIN

4.1 IF (SUPHORIZON-INFHORIZON) HORIZONMIN and

DEMHORIZON MAXDEMHOR

Then

BEGIN

- Increase the horizon by reading requests.
- For each request read, initialize a row in the INSERT matrix.
- Construct and modify the groups of neighboring requests.

END

4.2 Determine the next request to be inserted, d*.

4.3 Attempt phase 1 with request d*.

4.4 IF (d* has not been inserted) AND (there are still some available drivers).

THEN - Initialize a route for d*.

4.5 If d* has been inserted, carry out phase 2 with the route which received the request.

END.

5. POST-OPTIMIZATION

In this step, we attempt to minimize customer inconvenience (i.e. maximize the quality of service) while retaining the productivity level established in the previous step. For a given request (i), inconvenience is evaluated as a function of the deviation from the desired service time DTD(i), and of the excess ride time ERT(i). The user must indicate the relative importance of DTD(i) and ERT(i) using parameters associated with request priority.

For each request priority level, the user specifies :

PDTD(k) : The weight placed on deviations from the desired service time in the inconvenience function for a request of PRIORITE(k).

PERT(k) : The weight placed on excess ride time in the inconvenience function for a request of PRIORITE (k).

With these values, we define :

INCONV(i) : The inconvenience of request (i).

INCONV(i)=PDTD(PRIORITE(i))*DTD(i)+PERT(PRIORITE(i))*ERT(i).

By exchanging requests between routes, we attempt to reduce first the greatest inconvenience, then the second greatest inconvenience, and so on.

6. REFERENCES

1. Blais, J.-Y., Rousseau, J.-M., "HASTUS : An Evaluation Model for Drivers' Union Contract Negotiation in Transit Companies". Publication #163, Centre de recherche sur les transports, Université de Montréal. March 1980.

2. Bodin, L.D., Sexton, T.R., "The Multi-Vehicle Subscriber Dial-a-Ride Problem". Working paper MS/S 83-009. University of Maryland. February 1983.

3. Chapleau, L., Ferland, J., Lapalme, G., Rousseau, J.-M., "A Parallel Insert Method for the Capacitated Arc Routing Problem". Publication #234, Centre de recherche sur les transports, Université de Montréal. April 1982.

4. Haines, G.H., Jr., Wolf, R.N., "Alternative Approaches to Demand Responsive Scheduling Algorithms". Transportation Research A 16A, 1, 43-54, 1982.

5. Jaw, J.J., Odoni, A.R., Psaraftis, H.N., Wilson, N.H.W., "A Heuristic Algorithm for the Multi-Vehicle many too many Advance-Request Dial-a- Ride Problem". Working paper MIT-UMTA-82-3, Massachusetts Institute of Technology. June 1982.

6. Krolak, P.D., "TAXI - A Realistic Advance Reservation Dial-a-Ride Algorithm". Internal TSC Working paper, Report No. DOT-TSC-UM262-PM-82-35. October 1982.

7. Vaccaro, H., "Alternative Technique for Modeling Travel Distance". Masters Thesis in Civil Engineering, Massachusetts Institute of Technology. June 1974.

8. Rivard, G., "Construction des parcours des véhicules et des horaires des chauffeurs pour le transport des personnes handicapées". Publication #240, Centre de recherche sur les transports, Université de Montréal. May 1981.